Predicting Exercise Quality

Synopsis

It is now possible to inexpensively collect a large amount of data about personal activity using devices such as Jawbone Up, Nike FuelBand and Fitbit. These type of devices are parts of the quantified self movement – a group of enthusiasts who regularly take measurements about themselves to improve their health and find patterns in their behavior. In this project, the data from accelerometers on the belts, forearms, arms and dumbbells of 6 participants who were asked to perform barbell lifts correctly and incorrectly by 5 different ways, was analyzed. The aim was to predict the manner (classe) by which they did the exercise.

Data Analysis

Getting and Cleaning the Datasets

```
# Set the working directory.
# Load the required libraries.

library(dplyr)
library(caret)
library(randomForest)
library(e1071)
set.seed(1234)

# Download and read the datasets.

if(!file.exists("pml-training.csv") & !file.exists("pml-testing.csv")){
  gettrain <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
  gettest <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
  download.file(url = gettrain, destfile = "pml-training.csv")
  download.file(url = gettest, destfile = "pml-testing.csv")
  loadtrain <- read.csv("pml-training.csv", na.strings = c("NA", "", "#DIV/0!"))
  loadtest <- read.csv("pml-testing.csv", na.strings = c("NA", "", "#DIV/0!"))
} else{
  loadtrain <- read.csv("pml-training.csv", na.strings = c("NA", "", "#DIV/0!"))
  loadtest <- read.csv("pml-testing.csv", na.strings = c("NA", "", "#DIV/0!"))
}

# Remove the NA values

cleantrain <- loadtrain[, colSums(is.na(loadtrain)) == 0]
cleantest <- loadtest[, colSums(is.na(loadtest)) == 0]

# Remove the non-predictors.

cleantrain <- cleantrain[, -c(1:7)]
cleantest <- cleantest[, -c(1:7)]
```

```
dim(cleantrain)
```

```
## [1] 19622    53
```

```
dim(cleantest)
```

```
## [1] 20 53
```

Partitioning the Datasets

The training dataset was partitioned as 2 datasets. The training dataset had 60% and the test dataset had 40% of the data.

```
twoparts <- createDataPartition(cleantrain$classe, p = 0.6, list = FALSE)
training <- cleantrain[twoparts, ]
testing <- cleantrain[-twoparts, ]
```

Random Forest Model

The prediction was done using a customized random forest model.

```
custommod <- train(classe ~ ., data = training,
                   method = "rf", metric = "Accuracy",
                   preProcess = c("center", "scale"),
                   trControl = trainControl(method = "cv", number = 4, p = 0.6))
custommod
```

```
## Random Forest
##
## 11776 samples
##    52 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## Pre-processing: centered (52), scaled (52)
## Resampling: Cross-Validated (4 fold)
## Summary of sample sizes: 8833, 8830, 8832, 8833
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##    2    0.9867538  0.9832412
##   27    0.9886217  0.9856065
##   52    0.9839511  0.9796965
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 27.
```

Predicting and Comparing the Results Using the Testing Dataset.

```
predicttest <- predict(custommod, newdata = testing)
testing <- mutate(testing, classe = as.factor(classe))
levels(testing$classe)
```

```
## [1] "A" "B" "C" "D" "E"
```

```
confusionMatrix(predicttest, testing$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2231    4    0    0    0
##          B    0 1513    3    0    1
##          C    1    1 1361   10    2
##          D    0    0    4 1275    3
##          E    0    0    0    1 1436
##
## Overall Statistics
##
##                Accuracy : 0.9962
##                  95% CI : (0.9945, 0.9974)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
```

```
##                  Kappa : 0.9952
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9996   0.9967   0.9949   0.9914   0.9958
## Specificity            0.9993   0.9994   0.9978   0.9989   0.9998
## Pos Pred Value         0.9982   0.9974   0.9898   0.9945   0.9993
## Neg Pred Value         0.9998   0.9992   0.9989   0.9983   0.9991
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2843   0.1928   0.1735   0.1625   0.1830
## Detection Prevalence   0.2849   0.1933   0.1752   0.1634   0.1832
## Balanced Accuracy      0.9994   0.9980   0.9964   0.9952   0.9978
```

```
custommod$finalModel
```

```
##
## Call:
##  randomForest(x = x, y = y, mtry = param$mtry)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 27
##
##         OOB estimate of  error rate: 0.77%
## Confusion matrix:
##      A    B    C    D    E class.error
## A 3342    4    0    0    2 0.001792115
## B   24 2248    7    0    0 0.013602457
## C    0   13 2038    3    0 0.007789679
## D    0    0   24 1904    2 0.013471503
## E    0    1    4    7 2153 0.005542725
```

The accuracy is 99%. This is higher than 95%. There was sufficient confidence in the model to predict classe for the pml-testing dataset.

Validating the Model Using the pml-testing Dataset.

```
predict(custommod, newdata = cleantest)
```

```
##  [1] B A B A A E D B A A B C B A E E A B B B
## Levels: A B C D E
```